In [4]:
```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

In [5]:
```
data=pd.read_csv("/home/placement/Downloads/fiat500.csv")
```

In [6]:
```
data.describe()
```

Out[6]:

	ID	engine_power	age_in_days	km	previous_owners	lat	lon	price
count	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000
mean	769.500000	51.904421	1650.980494	53396.011704	1.123537	43.541361	11.563428	8576.003901
std	444.126671	3.988023	1289.522278	40046.830723	0.416423	2.133518	2.328190	1939.958641
min	1.000000	51.000000	366.000000	1232.000000	1.000000	36.855839	7.245400	2500.000000
25%	385.250000	51.000000	670.000000	20006.250000	1.000000	41.802990	9.505090	7122.500000
50%	769.500000	51.000000	1035.000000	39031.000000	1.000000	44.394096	11.869260	9000.000000
75%	1153.750000	51.000000	2616.000000	79667.750000	1.000000	45.467960	12.769040	10000.000000
max	1538.000000	77.000000	4658.000000	235000.000000	4.000000	46.795612	18.365520	11100.000000

In [7]:
```
data1=data.loc[(data.model=="lounge")]
data1
```

Out[7]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
6	7	lounge	51	731	11600	1	44.907242	8.611560	10750
7	8	lounge	51	1521	49076	1	41.903221	12.495650	9190
11	12	lounge	51	366	17500	1	45.069679	7.704920	10990
...	...	...	...	...	...	...	...	...	...
1528	1529	lounge	51	2861	126000	1	43.841980	10.515310	5500
1529	1530	lounge	51	731	22551	1	38.122070	13.361120	9900
1530	1531	lounge	51	670	29000	1	45.764648	8.994500	10800
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990

1094 rows × 9 columns

In [8]:
```
data1=data.drop(['lat','lon','ID'],axis=1)
```

In [9]:
```
data1
```

Out[9]:

	model	engine_power	age_in_days	km	previous_owners	price
0	lounge	51	882	25000	1	8900
1	pop	51	1186	32500	1	8800
2	sport	74	4658	142228	1	4200
3	lounge	51	2739	160000	1	6000
4	pop	73	3074	106880	1	5700
...	...	...	...	...	...	...
1533	sport	51	3712	115280	1	5200
1534	lounge	74	3835	112000	1	4600
1535	pop	51	2223	60457	1	7500
1536	lounge	51	2557	80750	1	5990
1537	pop	51	1766	54276	1	7900

1538 rows × 6 columns

In [10]:
```
data1.shape
```

Out[10]:
```
(1538, 6)
```

In [11]:
```
data1=pd.get_dummies(data1)
```

In [12]:
```
data1
```

Out[12]:

	engine_power	age_in_days	km	previous_owners	price	model_lounge	model_pop	model_sport
0	51	882	25000	1	8900	1	0	0
1	51	1186	32500	1	8800	0	1	0
2	74	4658	142228	1	4200	0	0	1
3	51	2739	160000	1	6000	1	0	0
4	73	3074	106880	1	5700	0	1	0
...	...	...	...	...	...	...	...	...
1533	51	3712	115280	1	5200	0	0	1
1534	74	3835	112000	1	4600	1	0	0
1535	51	2223	60457	1	7500	0	1	0
1536	51	2557	80750	1	5990	1	0	0
1537	51	1766	54276	1	7900	0	1	0

1538 rows × 8 columns

In [13]:
```
data1.shape
```

Out[13]:
```
(1538, 8)
```

In [14]:
```
y=data1['price']
x=data1.drop('price',axis=1)
```

In [ ]:
```

```

In [15]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test =train_test_split(x,y, test_size=0.33,random_state=42)#split data into trai
```

In [16]:
```
x_test.head(5)
```

Out[16]:

	engine_power	age_in_days	km	previous_owners	model_lounge	model_pop	model_sport
481	51	3197	120000	2	0	1	0
76	62	2101	103000	1	0	1	0
1502	51	670	32473	1	1	0	0
669	51	913	29000	1	1	0	0
1409	51	762	18800	1	1	0	0

In [17]:
```
x_train.shape
```

Out[17]:
```
(1030, 7)
```

In [18]:
```
y_train
```

Out[18]:
```
527      9990
129      9500
602      7590
331      8750
323      9100
        ...
1130    10990
1294     9800
860      5500
1459     9990
1126     8900
Name: price, Length: 1030, dtype: int64
```

In [19]:
```
y_test.head(5)
```

Out[19]:
```
481     7900
76      7900
1502    9400
669     8500
1409    9700
Name: price, dtype: int64
```

In [20]:
```
y_train.shape
```

Out[20]:
```
(1030,)
```

In [21]:
```
#for ridge regression
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge
alpha=[1e-15,1e-10,1e-8,1e-4,1e-3,1e-2,1,5,10,20,30]
ridge=Ridge()
parameters={'alpha':alpha}
ridge_regressor=GridSearchCV(ridge,parameters)
ridge_regressor.fit(x_train,y_train)
```

Out[21]:
```
▸   GridSearchCV
▸ estimator: Ridge
      ▸ Ridge
```

In [22]:
```
ridge_regressor.best_params_
```

Out[22]:
```
{'alpha': 30}
```

In [23]:
```
ridge=Ridge(alpha=30)
ridge.fit(x_train,y_train)
y_pred_ridge=ridge.predict(x_test)
```

In [24]:
```
from sklearn.metrics import mean_squared_error
Ridge_Error=mean_squared_error(y_pred_ridge,y_test)
Ridge_Error
```

Out[24]:
```
579521.7970897449
```

In [25]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred_ridge)
```

Out[25]:
```
0.8421969385523054
```

In [26]:

```
Results=pd.DataFrame(columns=['Price','Predicted'])
Results['Price']=y_test
Results['Predicted']=y_pred_ridge
Results=Results.reset_index()
Results['Id']=Results.index
Results.head(15)
```

Out[26]:

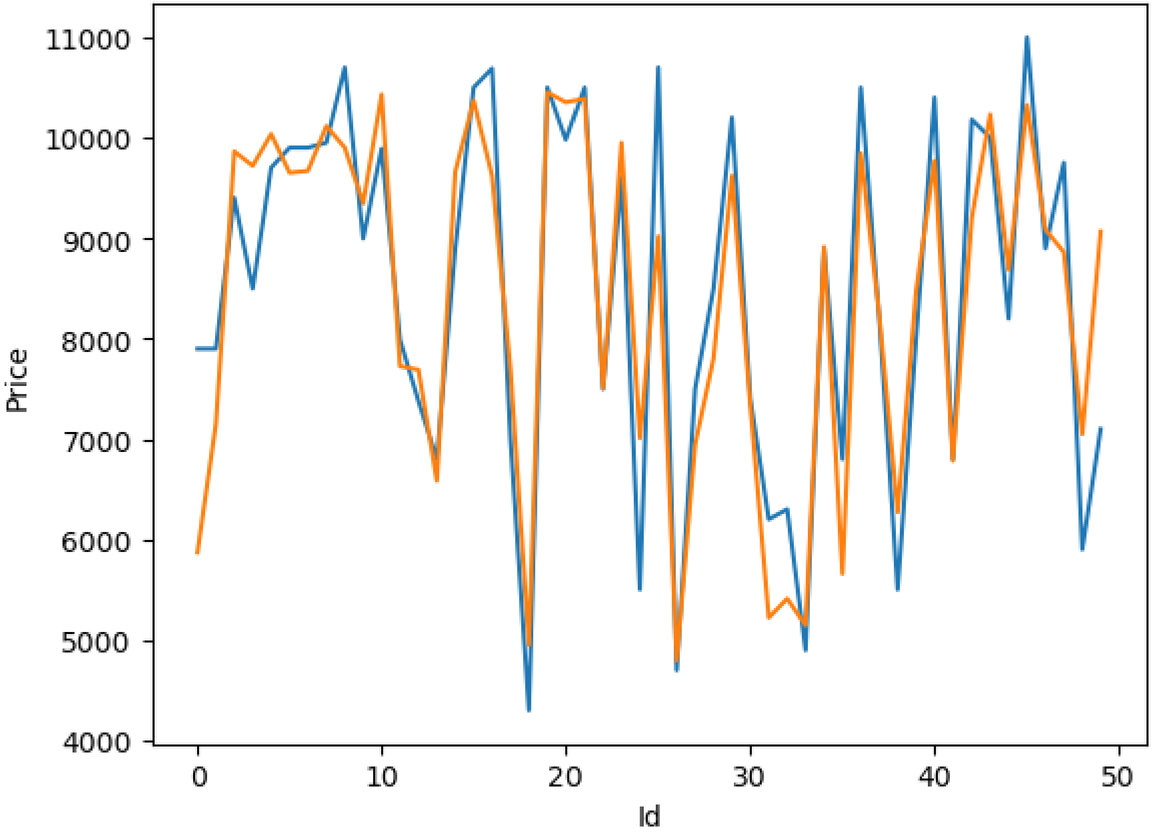
```
    index  Price     Predicted  Id
 0    481   7900   5869.741155   0
 1     76   7900   7149.563327   1
 2   1502   9400   9862.785355   2
 3    669   8500   9719.283532   3
 4   1409   9700  10035.895686   4
 5   1414   9900   9650.311090   5
 6   1089   9900   9669.183317   6
 7   1507   9950  10115.128380   7
 8    970  10700   9900.241944   8
 9   1198   8999   9347.080772   9
10   1088   9890  10431.237961  10
11    576   7990   7725.756431  11
12    965   7380   7691.089846  12
13   1488   6800   6583.674680  13
14   1432   8900   9659.240069  14
```

In [27]:

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.lineplot(x='Id',y='Price',data=Results.head(50))
sns.lineplot(x='Id',y='Predicted',data=Results.head(50))
plt.plot()
```

Out[27]:

```
[]
```



In [ ]: